IN THE UNITED STATES
PATENT AND TRADEMARK OFFICE

APPLICANTS:	Shaun P. Cooley
SERIAL NO.:	10/686,356
FILING DATE:	October 14, 2003
TITLE:	Countering Spam That Uses Disguised Characters
EXAMINER:	Tae K. Kim
GROUP ART UNIT:	2135
ATTY. DKT. NO.:	20423-08165

COMMISSIONER FOR PATENTS
P.O. BOX 1450
ALEXANDRIA, VA 22313-1450

DECLARATION OF FACT BY SHAUN P. COOLEY UNDER 37 C.F.R. § 1.131

I, Shaun P. Cooley, hereby declare the following:

1. I am the inventor of the invention described and claimed in U.S. Patent Application Serial No. 10/686,356 (hereinafter the "Subject Application"), entitled "Countering Spam That Uses Disguised Characters," and filed on October 14, 2003.

2. I actually reduced to practice the invention described and claimed, now pending in the Subject Application, in this country, before May 2, 2003. My prior actual reduction to practice are evidenced by the following:

a. Attached hereto as Exhibit A is a true and correct copy of an invention disclosure form, in which I disclosed my actual reduction to practice of the invention. I entered this invention disclosure form into Symantec's internal invention disclosure tracking system, which assigned tracking number 200308011345 to this invention disclosure form. Certain confidential information has been redacted from this invention disclosure form.

b. Attached hereto as Exhibit B is a true and correct copy of a printout from the source code control system containing the source code files indicated, including file "ColorLogic.h". Summaries of some of the files are also shown. The source code files had been checked into the source code control system before May 2, 2003. The printout also indicates that the source code files had been checked into the source code control system before May 2, 2003 (date redacted).

c. Attached hereto as Exhibit C is a true and correct copy of a printout of the source code file "ColorLogic.h" indicated in Exhibit B. This source code file had been checked into the source code control system before May 2, 2003.

d. Embodiments of the invention were constructed or performed prior to May 2, 2003 that met every element of the claims of the Subject Application. This is further evidenced by the following:

1. One such embodiment of the invention included executable code compiled from source code, the executable code stored on a computer readable storage medium corresponding to claim 16. Another such embodiment included an apparatus (e.g., computer) corresponding to claim 18 configured to execute the executable code. Another such embodiment included execution of the executable code to perform the method of claim 1.

2. The invention disclosure form of Exhibit A discusses each element of the claimed invention as recited in claims 1, 16, and 18. Section 10 of the invention disclosure form describes at least the following claimed elements: "countering spam that disguises characters within an electronic message;" "locating portions of the electronic message where a difference between foreground color and background color is negligible;" "determining whether at least one of the foreground color and the background color is a gray-scale color;" "responsive to at least one of the foreground color and the background color being a gray-scale color, deeming the difference between the colors to be negligible based on a comparison of

saturation and brightness values of the colors regardless of the hue values of the colors;" "deleting from the electronic message foreground characters from said portions, to form a redacted electronic message;" and "forwarding the redacted electronic message to a spam filter."

3. Section 5.A. of the invention disclosure form of Exhibit A demonstrates that the embodiment described in Section 10 of the invention disclosure form was completed before May 2, 2003 (date redacted).

4. The embodiments of the invention included or used the source code files listed in Exhibit B, including the source code file "ColorLogic.h".

5. The source code file "ColorLogic.h" shown in Exhibit C includes code for "locating portions of the electronic message where a difference between foreground color and background color is negligible, comprising: determining whether at least one of the foreground color and the background color is a gray-scale color; and responsive to at least one of the foreground color and the background color being a gray-scale color, deeming the difference between the colors to be negligible based on a comparison of saturation and brightness values of the colors regardless of hue values of the colors." See, e.g., the function "IsColorVisible()" in ColorLogic.h.

e. Embodiments of the invention were constructed or performed prior to May 2, 2003 that worked for its intended purpose. This is further evidenced by the following:

1. Section 9 of the invention disclosure form in Exhibit A mentions detecting spam as an intended purpose of the invention.

2. Section 10 of the invention disclosure form in Exhibit A further mentions that the invention enables detecting spam.

3. Section 5.A. of the invention disclosure form of Exhibit A demonstrates that the embodiment described in Sections 9 and 10 of the invention disclosure form was completed before May 2, 2003 (date redacted).

f. Testing was performed on the embodiments of the invention described above prior to May 2, 2003 and the testing successfully demonstrated that the invention worked for its intended purpose. This is further evidenced by the following:

1. The source code from the file "ColorLogic.h" shown in Exhibit C, along with other source code files, including the files listed in Exhibit B, was successfully compiled, executed, and tested prior to May 2, 2003. Specifically, this code was tested on multiple HTML files containing various foreground text colors and background colors. The code was determined to be able to successfully process the HTML files in accordance with the invention as described and claimed.

2. Section 5.E of the invention disclosure form of Exhibit A indicates that a "First Successful Version" of the embodiment described in Section 10 of the invention disclosure form has been retained, implying that the embodiment has been successfully tested.

3. The comment near the beginning of source code file "ColorLogic.h" shown in Exhibit C indicates that "the value ranges that are used are based on several weeks of testing hundreds of monitors." The "value ranges" refer to differences in values of hue, saturation, and brightness. The testing was used to determine optimal value ranges among other things.

4. The testing was performed prior to May 2, 2003.

g. The invention was recognized and appreciated prior to May 2, 2003. This is further evidenced by the following:

1. Section 9 of the invention disclosure form of Exhibit A mentions detecting spam as an intended purpose of the invention.

2. Section 10 of the invention disclosure form of Exhibit A further mentions that the invention enables detecting spam.

3. Section 5.A. of the invention disclosure form of Exhibit A demonstrates that the embodiment described in Sections 9 and 10 of the invention disclosure form was completed before May 2, 2003.

3. The claimed invention was therefore actually reduced to practice before May 2, 2003.

4. I hereby declare that all statements made herein to the best of my own knowledge are true and that all statements made on information and belief are believed to be true; that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under 18 U.S.C. § 1001; and that such willful statements may jeopardize the validity of the application or any patent issued thereon.

Shaun P. Cooley

March 2, 2009

Date

Exhibit A

PATENT PROPOSAL FORM

The purpose of this form is to gather information on potentially patentable technologies developed at Symantec and useful in its business. For any answer that requires more space than is provided, feel free to write additional information at the end of the form or on attached separate sheets or documents.

1. INVENTOR(S): (list the persons that participated in the development of the product/technology proposed to be patented):

A. Name Shaun Cooley Citizenship USA
Home Address [redacted]
Phone [redacted]

2. NAME OF MANAGER: Javed Khan

3.TITLE OF INVENTION:
Method for Detecting Nearly Invisible Text in Spam

4. CONCEPTION OF INVENTION: (provide whatever information is available):

A. Date of Conception [redacted]
B. Date of First Written [redacted]
Eng. Notebook No. _____ Page (s) _____
C. Date of First Oral Disclosure N/A_____

5. CONSTRUCTION AND TEST OF PRODUCT/TECHNOLOGY: (information regarding implementation of the product/technology; this does not have to be the full commercial version: provide information regarding the first or very early implementations that substantially accomplish the intended result):

A. Date Completed [redacted] Build No. _____
B. By Whom Made _____
C. Date of First Successful Test
D. Location of Code [redacted]
E. Has First Successful Version Been Retained? Yes
F. Anticipated Product: [redacted]

6. INVENTION RELATES TO:
This invention relates to improvements in spam detection by detecting an increasingly common trick used by spammers.

8. LIST OF PRIOR ART: (related printed publications, patents, reference materials or sources that you are aware of that describe prior art or form a basis for the product/technology proposed to be patented):

[redacted]

9. **OBJECT OF INVENTION:** (describe what new features, improvements over existing products/technology, or other advantages exist in the product/technology proposed to be patented as compared to the prior art described above):

Current statistical spam detection techniques are easily tricked in to believing that a message is not spam. If the sender of a spam message includes a large body of seemingly legitimate text, giving the text close to the same color as the background of the document, the legitimate words are still included in the statistical analysis. When this trick is used by a spammer, the spam message is typically 10-20 words long, while the legitimate text, included in the message, is typically 1000-2000 words long. Since the legitimate text is usually 100-200 times larger, the spam words have little influence on the final outcome of the statistical analysis.

10. **BRIEF DESCRIPTION:** (provide a brief description of the product/technology proposed to be patented, stressing the fundamental principles of the new idea from an engineering standpoint):

As previously discussed this invention enables a spam detection engine to recognize camouflaged text in the body of an email. The following components are required to accomplish this task:

1. An HTML parser that is capable of recognizing all methods of changing the foreground and background color in HTML. This includes, but is not limited to: style sheets, color attributes, background attributes, and inline styles.
2. A color comparison engine, capable of deciding if two colors are close enough to each other to make text unreadable.
3. A spam detection engine.

When an HTML email is to be classified by the spam detection engine, it is first passed through an HTML parser. The HTML parser tracks the foreground and background colors of text, throughout the document. When the foreground or background color changes, the difference in the two colors is calculated. If the difference in colors is deemed to be negligent, all text that uses the offending colors will be removed. When the HTML parser finishes removing camouflaged sections, the email continues on to the spam detection engine for processing.

[redacted] the color differences are calculated as follows:

1. Foreground (fg) and background (bg) colors are converted from Red/Green/Blue (RGB – *each value is in the range of 0-255*) values to Hue/Saturation/Brightness (HSB – *Hue is measured in degrees and thus has a range of 0-359, where 0 and 360 would be the same value; brightness and saturation are measured in percentages and have values of 0-100*) values.
2. The differences between the fg and bg colors are calculated.
    a. **Gray-Scale Values**

i. Since hue makes no difference in gray-scale colors, only the saturation and brightness are compared.
ii. If the saturation difference is less than 6 and the brightness difference is less than 4, the fg color is deemed invisible.

b. **Color Values**
i. If the hue difference is less than 6 and combined brightness and saturation difference is less than 14, the fg color is deemed invisible.

11. ALTERNATIVE EMBODIMENT(S), if any: (i.e., other ways that the same results could be accomplished, if you are aware of or have implemented any):

12. PUBLICATION:

A. Has a description of the product/technology been published?
Yes _____ No _____x_____
If Yes, Date of Publication _____
Title _____

B. Anticipated Publication
Yes _____ No _____
If Yes, Date of Publication_____
Title _____

13. SALE:

A. Was the product/technology ever offered for sale or sold?
Yes _____ No ___x_____
If Yes, Date of First Offer/Sale _____

B. Anticipated Sale
Yes _____ No ____x_____
If Yes, Date_____

14. OTHERS IN THE COMPANY FAMILIAR WITH PRIOR ART: (list the names of a few people in the Company that you feel are experts in the area of the product/technology proposed to be patented that may be able to provide additional information regarding similar products/technologies, related products/technologies and/or predecessors to the product/technology proposed to be patented):

15. FLOWCHARTS, PSEUDOCODE FOR SOFTWARE/PROCESS INVENTIONS:(If available, insert below, provide on a separate sheet or attach as a separate document)

Exhibit B

Change 523198 by scooley@cpd-scooley1-NAS on [redacted]

BayesHTMLFilter: Added more invisible ink detection and added domain-specific feature tracking
ColorLogic: dropped the borrowed method of color similarity testing, switch to RGB->HSB conversion and HSB comparisons
MailInfo: Now adding domain-specific features to the word bucket
WordBucket: added code to handle features that would have exceeded the WORD_MAX

Affected files ...

//depot/AntiSpam/trunk/src/asEngBay/BayesHTMLFilter.cpp#6 edit
//depot/AntiSpam/trunk/src/asEngBay/BayesHTMLFilter.h#6 edit
//depot/AntiSpam/trunk/src/asEngBay/ColorLogic.h#2 edit
//depot/AntiSpam/trunk/src/asEngBay/MailInfo.cpp#7 edit
//depot/AntiSpam/trunk/src/asEngBay/MailInfo.h#4 edit
//depot/AntiSpam/trunk/src/include/WordBucket.cpp#6 edit

Exhibit C

```
// Useful methods for converting and comparing colors
//////////////////////////////////////////////////////////////////////

#ifndef _COLOR_LOGIC_INCLUDE
#define _COLOR_LOGIC_INCLUDE

#include <math.h>

//////////////////////////////////////////////////////////////////////////////
// **  NOTE  **  NOTE  **  NOTE  **  NOTE  **  NOTE  **  NOTE  **  NOTE  **
//////////////////////////////////////////////////////////////////////////////
//
//
//
//
//
//
//
//
//
//
//
// This new method uses HSB space to determine if two colors are "too close"
// to each other be visible to on a CRT or LCD monitor.  The value ranges
// that are used are based on several weeks of testing hundreds of monitors.
//
//////////////////////////////////////////////////////////////////////////////
// **  NOTE  **  NOTE  **  NOTE  **  NOTE  **  NOTE  **  NOTE  **  NOTE  **
//////////////////////////////////////////////////////////////////////////////

namespace ColorLogic
{
        // Hue/Saturation/Brightness struct
        typedef struct tagHSB
        {
                int nHue;                       // Degree (0-360)
                int nSaturation;        // Percentage (0-100)
                int nBrightness;        // Percentage (0-100)
        } HSB;

        // Round a double to the given precision
        // Used in RGB->HSB conversion method
        double DblRound(double dValue, int dPrecision)
        {
                static const double dBase = 10.0f;
                double dComplete5, dComplete5i;

                dComplete5 = dValue * pow(dBase, (double)(dPrecision + 1));

                if(dValue < 0.0f)
                        dComplete5 -= 5.0f;
                else
                        dComplete5 += 5.0f;

                dComplete5 /= dBase;
                modf(dComplete5, &dComplete5i);

                return dComplete5i / pow(dBase, (double)dPrecision);
        }
```

```
// returns the difference between the min and the max
int minmax(int i1, int i2)
{
        return max(i1, i2) - min(i1,i2);
}

// Converts the given RGB color to Hue/Saturation/Luminance
// Note: Photoshop seems to floor values instead of rounding...
//       rounding is more accurate
void RGB_to_HSB(COLORREF crRGB, HSB& hsb)
{
        WORD wRed = GetRValue(crRGB);
        WORD wGreen = GetGValue(crRGB);
        WORD wBlue = GetBValue(crRGB);

        // Find the min and max RGB values
        WORD wMax = max(wRed, max(wGreen, wBlue));
        WORD wMin = min(wRed, min(wGreen, wBlue));

        // Calculate the brightness
        hsb.nBrightness = (int)DblRound((((double)wMax * 100) / 255), 0);

        // If this is grey we are done
        if(wMax == wMin)
        {
                hsb.nHue = 0;
                hsb.nSaturation = 0;
        }
        else
        {
                // Calculate the saturation
                hsb.nSaturation = (int)DblRound((((double)100 * (wMax -
wMin)) / wMax), 0);

                // Calculate the hue
                double dDiff = wMax - wMin;
                double dR = (wMax - wRed) / dDiff;
                double dG = (wMax - wGreen) / dDiff;
                double dB = (wMax - wBlue) / dDiff;
                double dHue = 0;
                if(wRed == wMax)
                        dHue = dB - dG;
                else if(wGreen == wMax)
                        dHue = 2 + dR - dB;
                else if(wBlue == wMax)
                        dHue = 4 + dG - dR;

                hsb.nHue = (int)DblRound((dHue * 60) + 360, 0) % 360;
        }
}

bool IsColorVisible(COLORREF crFG, COLORREF crBG)
{
        // RGB->HSB conversions
        HSB hsbFG, hsbBG;
        RGB_to_HSB(crFG, hsbFG);
        RGB_to_HSB(crBG, hsbBG);

        bool bVisible = true;
```

```
                // the hue is in degrees and can wrap around, so we find
                // the shortest distance between the two colors
                int nHueDiff = abs(hsbFG.nHue - hsbBG.nHue);
                if(nHueDiff > 180)
                        nHueDiff = abs(nHueDiff - 360);

                // Saturation and Brightness differences are checked together
                int nSDiff = abs(hsbFG.nSaturation - hsbBG.nSaturation);
                int nBDiff = abs(hsbFG.nBrightness - hsbBG.nBrightness);
                int nSBDiff = nBDiff + nSDiff;

                // Handle B/W colors differently
                // (since the HUE makes no difference)
                if(max(hsbFG.nSaturation, hsbBG.nSaturation) <= 5
                        && nSDiff < 6)
                {
                        if(nBDiff < 4)
                                bVisible = false;
                }
                else
                {
                        // If the FG hue is within 5 of the BG hue...
                        if(nHueDiff <= 5
                                && nSBDiff <= 13)
                        {
                                bVisible = false;
                        }
                }

                return bVisible;
        }
} // namespace ColorLogic
#endif // _COLOR_LOGIC_INCLUDE
```